

AI Tools for Jakarta EE



Empowering Developers with Next-Gen AI Capabilities

Gaurav Gupta,
@jGauravGupta, github.com/jGauravGupta
Senior Software Engineer @ Payara Service Limited,
Apache Committer, NetBeans Dream Team Member,
Jeddict Creator <https://jeddict.github.io/>

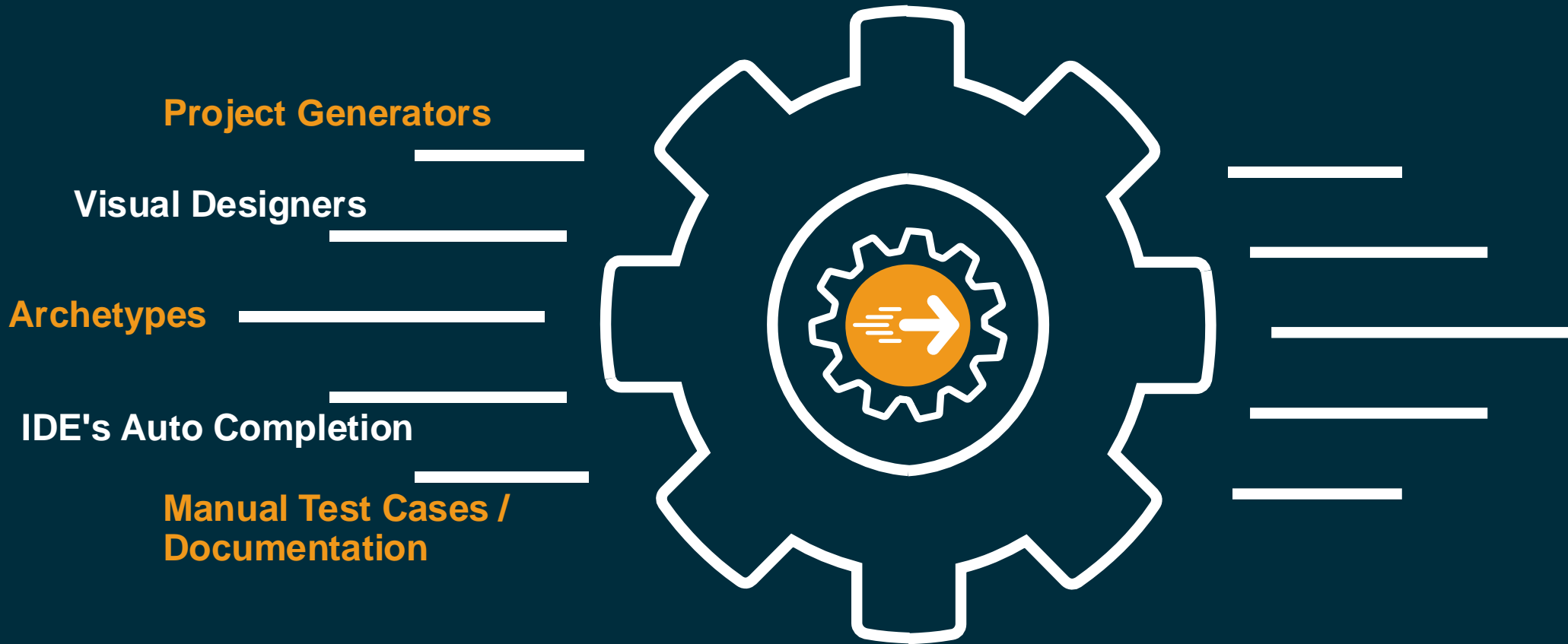
Agenda

Creating SFSCON Conference App Using AI Tools

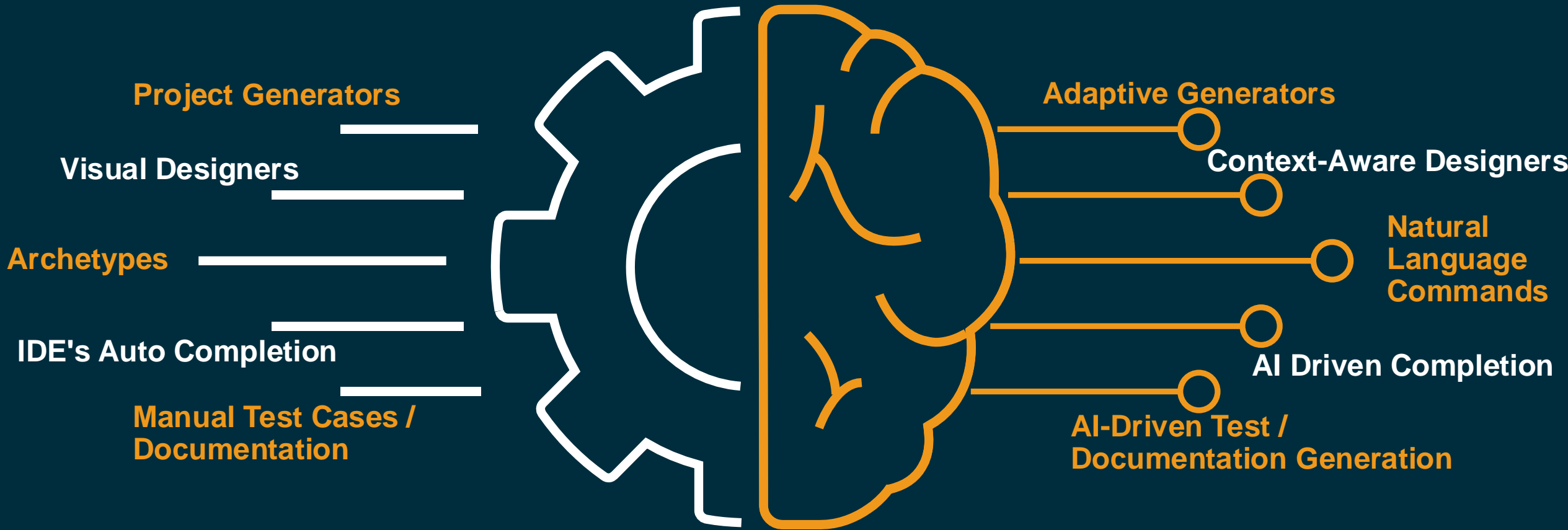
Disclaimer

*The application generated in this demonstration utilizes **AI-powered tools** and serves as a foundational prototype. AI-generated code may contain compilation errors, inconsistencies, and is inherently variable with each generation attempt. This code is not production-ready and should be treated as a preliminary draft. Comprehensive review, refinement, and testing by experienced developers are essential to ensure functionality, performance, security, and adherence to production standards.*

Traditional Development



AI-Powered Development

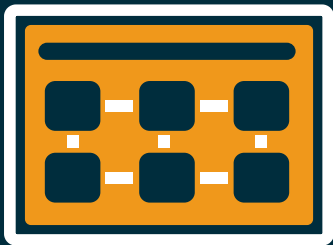


Introducing Payara Starter

<https://start.payara.fish/>



MicroProfile



ER Diagram Designer

The screenshot shows the "Project Description" step of the Payara Starter wizard. It includes a progress indicator at the top left, a "Build" section with radio buttons for "Maven" (selected) and "Gradle", and input fields for "Group ID" (fish.payara), "Artifact ID" (sfscon), and "Version" (0.1-SNAPSHOT). A "Next" button is located below the version field. A vertical progress list on the right side of the form shows the following steps: Jakarta EE, Payara Platform, Project Configuration, MicroProfile, Deployment Options, ER Diagram Designer, and Security Configuration. A "Generate" button is at the bottom of the form.



Secured

ER Diagram Designer

Zero Domain Knowledge

Source

SFSCON

Live Preview



```
1 erDiagram
2   CONFERENCE ||--o{ ATTENDEE : has
3   CONFERENCE {
4     int conferenceID PK
5     string name
6     date date
7     string location
8   }
9   ATTENDEE {
10    int attendeeID PK
11    string firstName
12    string lastName
13    string email
14  }
```

Create new Diagram

CONFERENCE		
int	conferenceID	PK
string	name	
date	date	
string	location	



has

ATTENDEE		
int	attendeeID	PK
string	firstName	
string	lastName	
string	email	

Tell us how you want to change the diagram (e.g., 'Add customer address') and press Enter

Navigation icons: left arrow, right arrow, plus, minus, zoom out

ER Diagram Designer

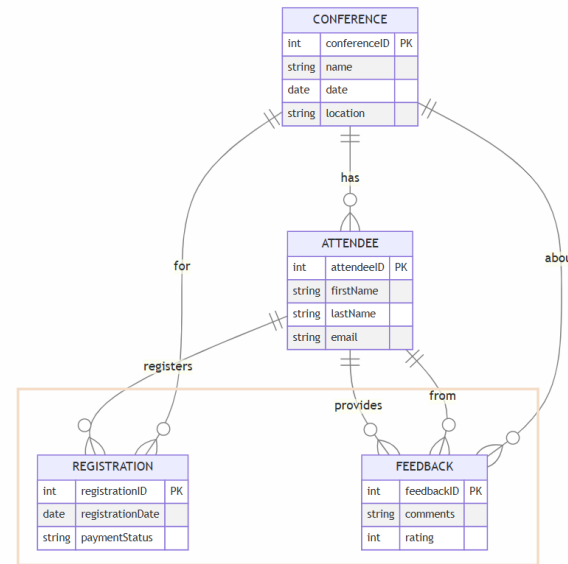
Natural Language Command

Source SFSCON

```
1 erDiagram
2 CONFERENCE ||--o{ ATTENDEE : has
3 CONFERENCE {
4     int conferenceID PK
5     string name
6     date date
7     string location
8 }
9 ATTENDEE {
10     int attendeeID PK
11     string firstName
12     string lastName
13     string email
14 }
15 ATTENDEE ||--o{ REGISTRATION : registers
16 REGISTRATION {
17     int registrationID PK
18     date registrationDate
19     string paymentStatus
20 }
21 REGISTRATION }o--|| CONFERENCE : for
22 ATTENDEE ||--o{ FEEDBACK : provides
23 FEEDBACK {
24     int feedbackID PK
```

Natural Language Command

Live Preview



Tell us how you want to change the diagram (e.g., 'Add customer address') and press Enter



ER Diagram Designer

Context Awareness

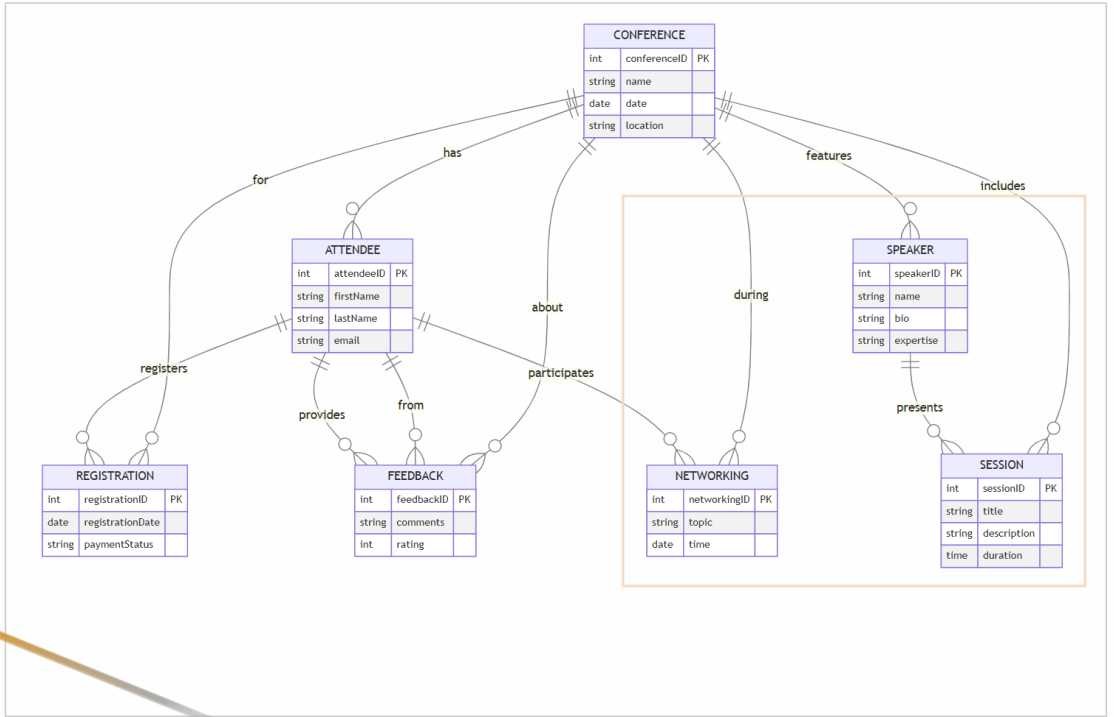
Source

Live Preview ✕

```
1 erDiagram
2 CONFERENCE ||--o{ ATTENDEE : has
3 CONFERENCE {
4     int conferenceID PK
5     string name
6     date date
7     string location
8 }
9 ATTENDEE {
10     int attendeeID PK
11     string firstName
12     string lastName
13     string email
14 }
15 ATTENDEE ||--o{ REGISTRATION : registers
16 REGISTRATION {
17     int registrationID PK
18     date registrationDate
19     string paymentStatus
20 }
21 REGISTRATION }o--|| CONFERENCE : for
22 ATTENDEE ||--o{ FEEDBACK : provides
23 FEEDBACK {
24     int feedbackID PK
```

Mermaid Diagram

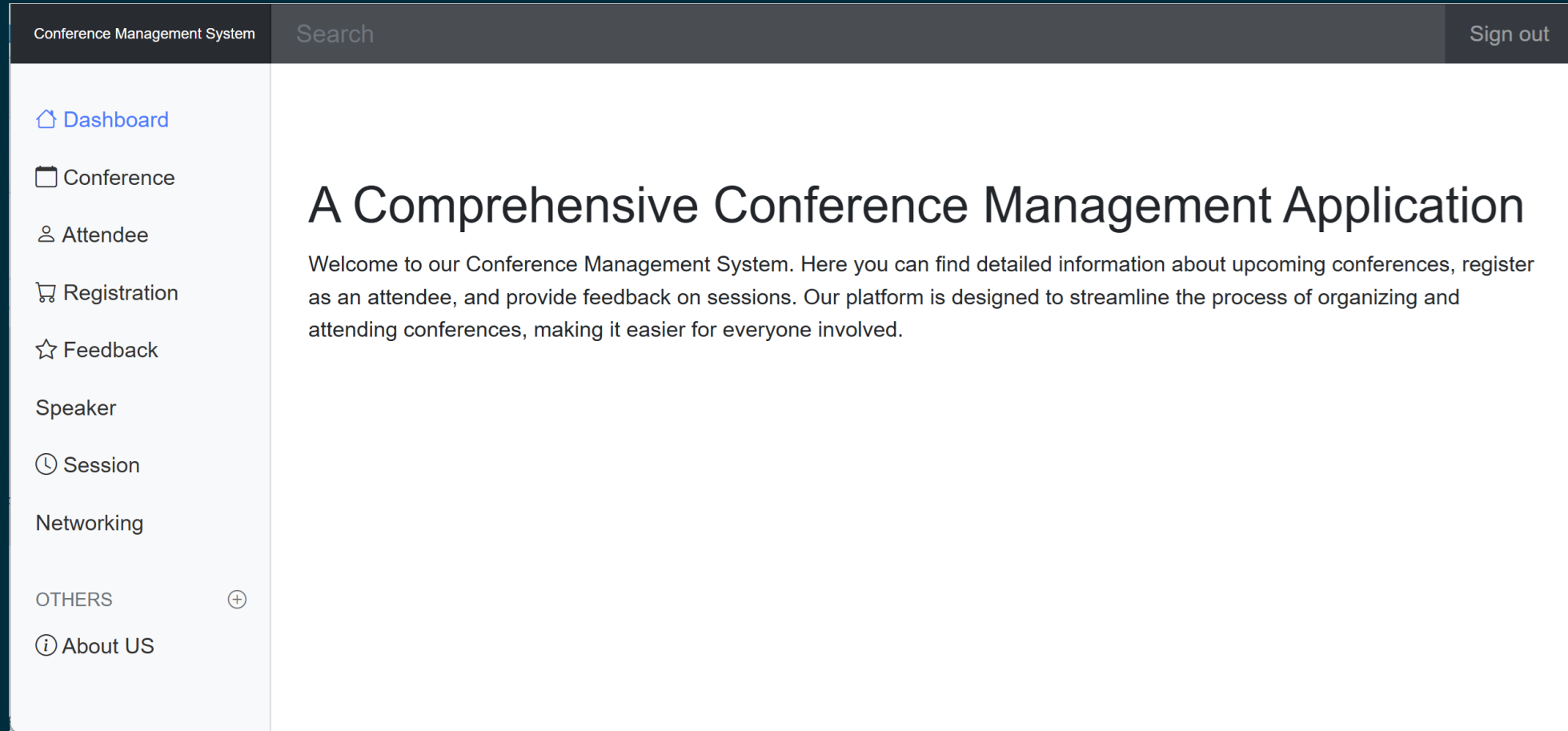
Enlarge Diagram



Tell us how you want to change the diagram (e.g., 'Add customer address') and press Enter

Navigation controls: left arrow, right arrow, plus, minus, and zoom in/out icons.

Adaptive Generator for Jakarta EE



The screenshot shows a web application interface for a Conference Management System. The top navigation bar includes the text "Conference Management System", a search input field, and a "Sign out" button. A left sidebar contains a list of menu items: "Dashboard" (with a home icon), "Conference" (with a calendar icon), "Attendee" (with a person icon), "Registration" (with a ticket icon), "Feedback" (with a star icon), "Speaker", "Session" (with a clock icon), "Networking", "OTHERS" (with a plus icon), and "About US" (with an information icon). The main content area features a large heading "A Comprehensive Conference Management Application" and a paragraph of introductory text.

Conference Management System Search Sign out

- Dashboard
- Conference
- Attendee
- Registration
- Feedback
- Speaker
- Session
- Networking
- OTHERS +
- About US

A Comprehensive Conference Management Application

Welcome to our Conference Management System. Here you can find detailed information about upcoming conferences, register as an attendee, and provide feedback on sessions. Our platform is designed to streamline the process of organizing and attending conferences, making it easier for everyone involved.

Generated Application with Payara Starter

Conference Management System Search Sign out

- Dashboard
- Conference
- Attendee
- Registration
- Feedback
- Speaker
- Session
- Networking
- OTHERS +
- About US

Conference

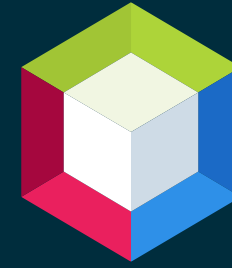
Represents a conference event.

[Add Conference](#)

Conference ID	Name	Date	Location	Actions
1	SFSCON 2024	2024-11-08	Bozen Italy	Edit Delete



Jeddict AI Assistant for Apache NetBeans IDE



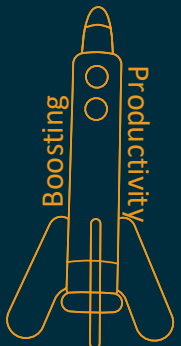
- Completely Free AI Assistant
- Project / Package context-aware Chat
- AI-Driven inline Code Completion
- SQL inline completion
- Context-aware database interactions
- Create REST Endpoints
- Support for multiple LLM Provider

Jeddict AI Assistant Intelligent Code Completion

```
4 import jakarta.persistence.EntityManager;
5 import jakarta.inject.Inject;
6 import fish.payara.domain.Speaker;
7
8 @Dependent
9 public class SpeakerService extends AbstractService {
10
11     @Inject
12     private EntityManager em;
13
14     @Override
15     protected EntityManager getEntityManager() {
16         return em;
17     }
18
19     public SpeakerService() {
20         super(Speaker.class);
21     }
22
23     // send email to speaker
24
25
26
27
28 }
```

Method to send email to a speaker
Uses the JavaMail API to send an email, which is essential for notifying speakers.

- private static final String DEFAULT_EMAIL_SUBJECT = "Important Speaker Update"
- public List<Speaker> getAllSpeakers() { return em.createQuery("select s from Speaker s").getResultList(); }
- public void sendEmailToSpeaker(Speaker speaker, String subject, String message) { ... }
- public void updateSpeaker(Speaker speaker) { em.merge(speaker); }
- SpeakerService(EntityManager em, Class<E> entityClass) - generate
- clone() - override
- count() - override
- create(Speaker entity) - override
- edit(Speaker entity) - override



Jeddict AI Assistant

Context Aware Variable / Method Naming

FOOBAR

data/x2

data/x

foobar'

foo 22'

temp123'

```
28 @Path("/api/speaker")
29 public class SpeakerResource {
30
31     private static final Logger LOG = Logger.getLogger(SpeakerResource.class.get
32
33     @Inject
34     private SpeakerService speakerService;
35
36     @Inject
37     private ConferenceService conferenceService;
38
39     private static final String ENTITY_NAME = "speaker";
40
41     /** POST : Create a new speaker ...9 lines */
42
43     @POST
44     @Consumes(MediaType.APPLICATION_JSON)
45     @Produces(MediaType.APPLICATION_JSON)
46     public Response createSpeaker(Speaker speaker) throws URISyntaxException {
47
48         LOG.log(Level.INFO, "save Speaker : {0}", speaker);
49
50         if (speaker.getConference() != null) {
51             speakerService.enrollSpeaker(speaker.getConference().getConferenceID(), speaker);
52         } else {
53             speakerService.insertSpeaker(speaker);
54         }
55         speakerService.persistSpeaker(speaker);
56         speakerService.processSpeakerCreation(speaker);
57         return Response.status(201).entity(speakerService.registerSpeaker(speaker)).header("Location", speakerService.saveSpeaker(speaker).getConference().getConferenceID().toString());
58     }
59
60     private SpeakerService speakerService;
61     private ConferenceService conferenceService;
62     private static final String ENTITY_NAME = "speaker";
63 }
```

data | 22

data | x?

foobar x2

temp|22

temp122

temp123

Jeddict AI Assistant Log Message / JavaDocs

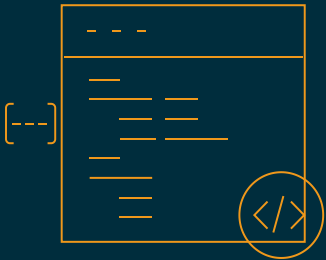
The screenshot shows an IDE interface with a project explorer on the left and a code editor on the right. The project explorer shows a project named 'sfscon' with several sub-packages, including 'fish.payara.domain' and 'fish.payara.resource'. The code editor displays the 'SpeakerResource.java' file, which contains a REST endpoint for enrolling a speaker. A log message is highlighted in the code, and a tooltip is shown over it, listing several log messages related to the speaker enrollment process.

```
28 @Path("/api/speaker")
29 public class SpeakerResource {
30
31     private static final Logger LOG = Logger.getLogger(SpeakerResource.class.get
32
33     @Inject
34     private SpeakerService speakerService;
35
36     @Inject
37     private ConferenceService conferenceService;
38
39     private static final String ENTITY_NAME = "speaker";
40
41     /** POST : Create a new speaker ...9 lines */
42
43     @POST
44     @Consumes(MediaType.APPLICATION_JSON)
45     @Produces(MediaType.APPLICATION_JSON)
46     public Response enrollSpeaker(Speaker speaker) throws URISyntaxException {
47         LOG.log(Level.FINE, "REST request to save Speaker : {}", speaker);
48
49         if (speaker.getCon
50             speaker.setCon
51         } else {
52             speaker.setCon
53         }
54         speakerService.cre
55         return HeaderUtil.
56             ENTITY_NAM
57             .entity(sp
```

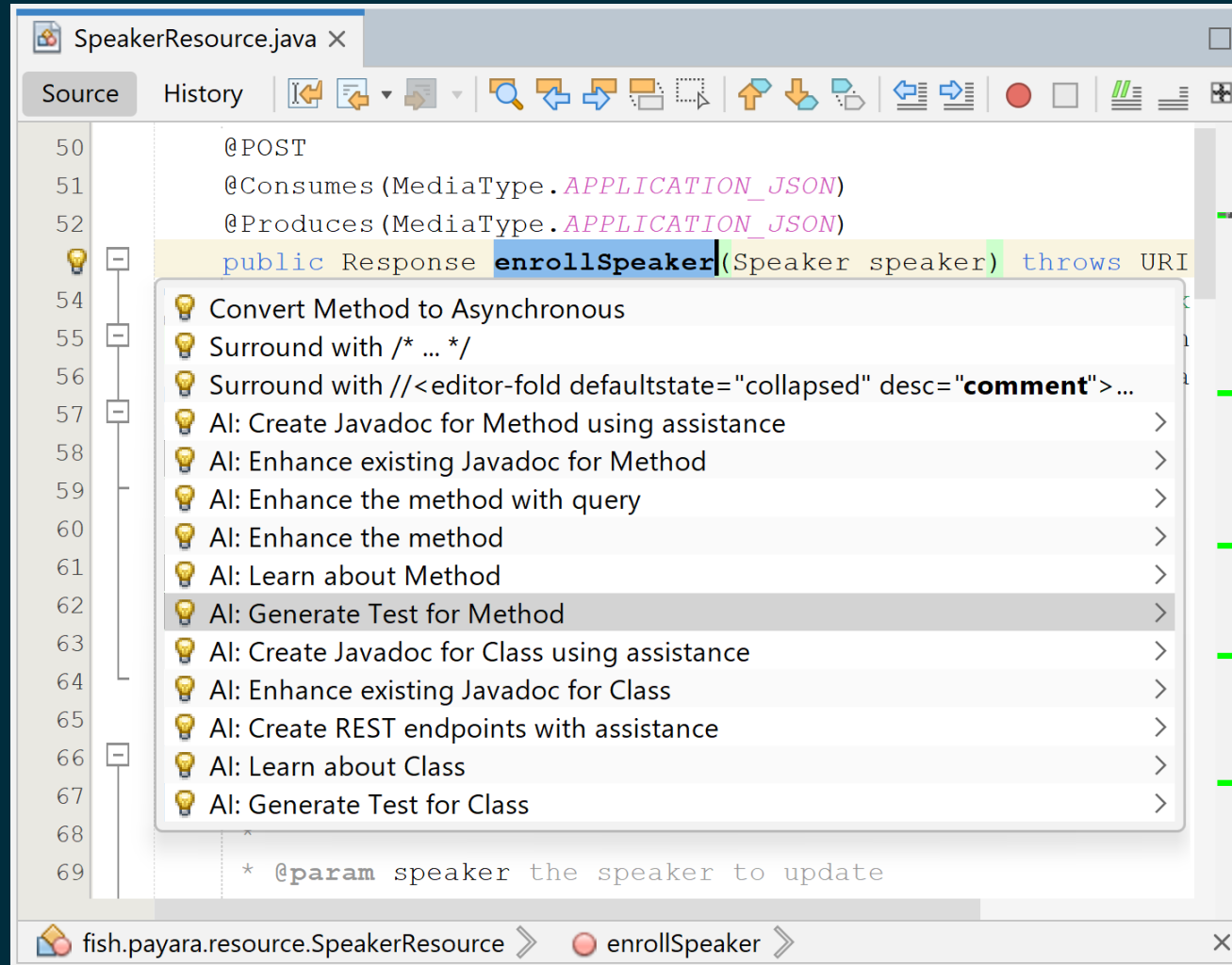
Log messages in the tooltip:

- Attempting to save Speaker data: {}
- Initiating REST request to save Speaker: {}
- Logging Speaker creation request: {}
- Processing request to create Speaker: {}
- Request received to enroll Speaker: {}
- Request to save Speaker entity: {}
- Saving Speaker information: {}
- Storing new Speaker details: {}

Java Docs



Jeddict AI Assistant Test case generation



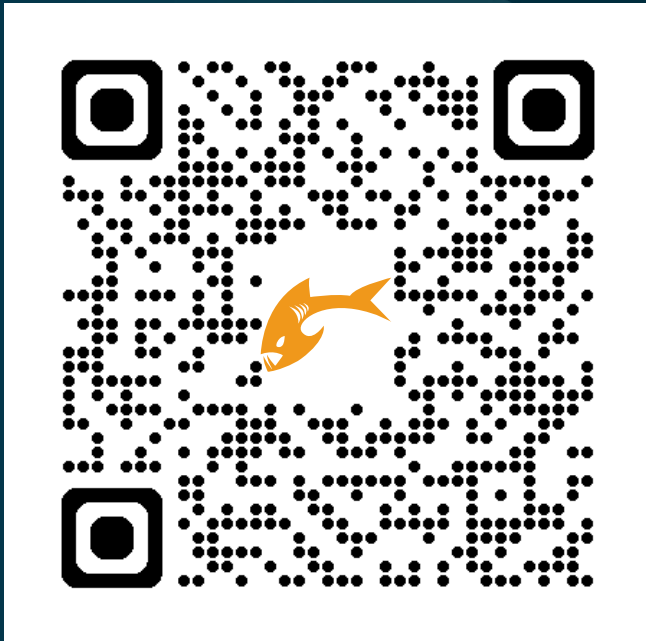
```
50 @POST
51 @Consumes(MediaType.APPLICATION_JSON)
52 @Produces(MediaType.APPLICATION_JSON)
53 public Response enrollSpeaker(Speaker speaker) throws URI
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69 * @param speaker the speaker to update
```

- Convert Method to Asynchronous
- Surround with /* ... */
- Surround with //<editor-fold defaultstate="collapsed" desc="comment">...
- AI: Create Javadoc for Method using assistance >
- AI: Enhance existing Javadoc for Method >
- AI: Enhance the method with query >
- AI: Enhance the method >
- AI: Learn about Method >
- AI: Generate Test for Method >
- AI: Create Javadoc for Class using assistance >
- AI: Enhance existing Javadoc for Class >
- AI: Create REST endpoints with assistance >
- AI: Learn about Class >
- AI: Generate Test for Class >



Thank You

<https://start.payara.fish>



<https://jeddickt.github.io>

