WEEK 1 - PROGRAMMING THINKING AND METHODS

Subject		Key Activities	Outcome
Day 1	Opening and team building	1. Session: training plan sharing 2. Activity: ice-breaking 3. Session: learning pyramid 4. Session: PDCA introduction 5. Demo & practice: concept map 6. Demo & practice: ORID 7. Session: standup meeting	1. Can understand the goal of the training 2. Can be aware of the value of self-learning, and can't wait to try 3. Can use concept map to self-learning and summarization 4. Can write diary based on ORID 5. Can understand the target of standup meeting and know how to do standup
Day 2	Tasking	1. Demo & practice how to clarify requirements 2. Demo & practice drawing task diagram and transform requirements into programming tasks 3. Demo & practice principles of Naming 4. Coding exercise: Multiplication table & Pos Machine 5. Demo & practice: standup meeting 6. Demo & practice: Git commit	1. Can clarify requirement 2. Can use the Tasking techniques to break down complex process into small pieces 3. Can understand the better naming practice 4. Can apply tasking techniques to improve the programming 5. Can practice version control with git in common scenario 6. Can organize and run the standup meeting
Day 3	Java OO programming	1. Demo & practice how to practice code review 2. Demo & practice with java collection 3. Session: OOP introduction 4. Demo & practice encapsulation, composition and polymorphism 5. Coding exercise: OO step-by-step	1. Can use encapsulation, composition and polymorphism related syntax 2. Can understand the principles of object-oriented design 3. Can understand java stream api
Day 4	TDD and practicing	1. Demo & practice the way we design & write unit test 2. Demo & practice Test-Driven Development process 3. Coding exercise: Mars Rover 4. Demo & practice: fix bugs or errors	1. Can understand the values & principles of unit testing 2. Can design & write unit test 3. Can understand the values & principles of Test-Driven Development 4. Can follow Test-Driven Development process 5. Can understand how to fix bug or errors
Day 5	Advanced TDD with OO	1. Demo & practice how to clarify requirements 2. Demo & practice how to sort out ACs 3. Demo & practice how TDD builds test cases 4. Coding exercise: Parking lot 5. Retrospective introduction and practice	1. Can sort out ACs based on requirements. 2. Can use TDD to build test cases 3. Can grasp the benefits of development self-test and develop testing habits 4. Can do retrospective meeting

WEEK 2 - JAVA FOUNDATION AND API BUILDING

Subject		Key Activities	Outcome
Day 1	Refactoring to clean code	1. Trainees team presentation: OO design pattern 2. Demo & practice refactoring 3. Demo & practice identifying code smells 4. Demo & practice refactoring techniques 5. Coding Practice: Refactoring legacy code to readable & maintainable code	1. Can clarify requirement before implementation 2. Can understand the purpose of refactoring and the value of clean code 3. Can reconstruct more readable and maintainable code independently or through teamwork 4. Can understand how to identify code smell, and use refactoring techniques to safely modify and optimize the code
Day 2	RESTful api design	1. Session: HTTP Basic 2. Demo & practice how to understand HTTP request and response message 3. Demo & practice how to understand JSON to express data 4. Session: RESTful API 5. Demo & practice how to design RESTful API 6. Coding exercise: RESTful API design 7. Demo & practice how to develop RESTful API and verify with Postman 8. Coding exercise: employee	1. Can understand HTTP methods, HTTP status code and HTTP request and response message 2. Can understand JSON and use it as request body and response body 3. Can choose the appropriate method and design APIs based on REST principles from a consumer perspective 4. Can implement RESTful APIs including GET, POST, PUT, DELETE
Day 3	Spring backend development	1. Demo & practice how to write API testing to verify RESTful API 2. Coding exercise: api test for employee 3. Demo & practice how to use TDD and refactor RESTful API code to multi-layer structure 4. Coding exercise: TDD and refactor to multi-layer structure 5. Demo & practice how to handle exception globally	1. Can understand the latest back-end development specifications 2. Can write API tests for http requests 3. Can understand the value of layered architecture 4. Can implement the layered architecture 5. Can use TDD with spring boot
Day 4	ORM & database	1. Demo & practice how to use ORM to map object to database tables 2. Demo & practice how to database CRUD operations 3. Demo & practice how to use database migration tool to manage database version 4. Coding exercise: database CRUD operations 5. Coding exercise: database migration	1. Can use spring JPA to map object to database tables 2. Can implement database CRUD operations 3. Can use flyway to manage the table structure and table version in the database
Day 5	Cloud native introduction	1. Cloud native basic knowledge Introduction 2. Trainees session with cloud native topics: microservice, container and DevOps 3. Coach introduce the concepts, history and the key points of cloud native 4. Coach introduce the 12 factors about cloud native 5. Retrospective activity	1. Can have the basic understanding with cloud native related concepts 2. Can understand the key points of microservice, container, DevOps 3. Can know the 12 factors about cloud native

WEEK 3 - FRONTEND FOUNDATION AND STACK-BASED DEVELOP

Subject		Key Activities	Outcome
Day 1	Frontend recap and React introduction	1. Session: retrospective about JavaScript, including array api, arrow function, spread operator, template literals, destructuring objects, object property shorthand 2. Session: introduction on react 3. Demo & practice how to design the component based on UI page 4. Coding exercise: react component development using props to passing data between parent and child components	1. Can understand and clarify the requirement base on UI page 2. Can split the UI page into different components 3. Can implement single react component 4. Can implement data passing using component props
Day 2	React Redux	Session: why react redux Demo & practice how to use react redux Coding exercise: refactoring application to react redux Demo & practice how to design and use react redux from beginning Coding exercise: todo list with redux	1. Can understand component lifecycle based on the data passing 2. Can use the react hooks api to manage data 3. Can implement data update using effect hook api 4. Can implement conditional rendering
Day 3	React Redux with API	1. Session: react router 2. Session: promise and axios 3. Demo & practice how to use react router 4. Demo & practice how to call api in frontend 5. Demo & practice how to use ant design to simplify frontend development 6. Coding exercise: todo list with ant design	1. Can understand the value of react redux 2. Can understand the principles and working mechanisms of react redux 3. Can use react redux to build simple front-end applications
Day 4	Frontend & backend integration	1. Demo & practice how to integrate frontend with backend 2. Session: CORS 3. Demo & practice: integration between frontend and backend 4. Coding exercise: todo list integration 5. Trainees session with agile activity session: MVP, User Journey, User Story, Elevator Speech 6. Retro	1. Can use react router 2. Can use Ant Design to design UI 3. Can send request from the frontend 4. Can solve CORS 5. Can integrate both frontend and backend
Day 5	Scrum activities	1. Simulation project introduction 2. Team roles introduction 3. Agile project introduction, including elevator speech, user journey, MVP, story card, Kanban board, IPM, CI/CD 4. Retrospective activity	1. Can know how to practice agile activities 2. Can know how to teamwork in agile 3. Can know the best practice for agile 4. Can use User Journey to analyze requirements 5. Can identify the MVP of the product 6. Can use Kanban board to manage story cards 7. Can setup CI

WEEK 4 - FULL STACK AGILE PROJECT FOR PRODUCTION PROCESS

Subject		Key Activities	Outcome
Day 1-4	Agile project building *Team iterations*	1. Read the requirement and kick-off 2. Design the agile kanban for visualizing the delivery process 3. Design the user journey for requirement context and solution 4. Design and write the user story 5. Discuss and make the Iteration and release plan 6. Design the API 7. Design the domain model 8. Implementation the API via TDD 9. Refactor the smell code to clean code 10. Write the API test to confirm they meet requirement 11. Build the UI with react and get/post data from backend API 12. Deploy to cloud platform 13. Summary the knowledge and skills and design own showcase report	1. Can understand the develop process in production environment 2. Can know how to use the skills, tools and knowledge in a whole project 3. Can validate the knowledge, skills they have handled or still need to learn and improve 4. Can apply the trained knowledge comprehensively and build a foundation to develop a APP with front and back separation
Day 1-4	Showcase for iteration target *Showcase schedule*	1. Showcase for all stakeholders 2. Showcase time is 10am per day 3. Showcase process should be a user view scenario at least 4. Showcase content must be relate to iteration target	1. Team has their own exact target to deliver and make output 2. Teams are able to know what is user view and want is outcome for a product
Day 5	Ending and close	1. Showcase includes whole agile project and what they learned in this bootcamp 2. Review all knowledge and concepts in this bootcamp 3. Trainee personal learn and action plan 4. Issuing certification for every trainees	