This is a Java class named `VoterController`, which serves as the controller for handling HTTP requests related to voters in the application. It uses Spring's `@RestController` and `@RequestMapping` annotations to define the API endpoints. Let's go through the code and explain each part:

1. `@RestController`: This annotation is used to mark the class as a controller that handles HTTP requests and returns responses as JSON data.

2. `@RequestMapping("/auth/p2")`: This annotation specifies the base path for all the endpoints defined in this controller. All the endpoints defined in this class will start with "/auth/p2".

3. `@RequiredArgsConstructor`: This Lombok annotation automatically generates a constructor with required arguments for the class fields. In this case, it will create a constructor with arguments for `service` and `voterAuthenticationService`.

4. `private final VoterService service;`: This field holds an instance of `VoterService`, which is presumably a service class that provides methods to interact with the voter data.

5. `private final VoterAuthenticationService voterAuthenticationService;`: This field holds an instance of `VoterAuthenticationService`, which is presumably a service class responsible for voter authentication and registration.

6. `@GetMapping("/voterList/All")`: This annotation maps a GET request to "/auth/p2/voterList/All" to the `getVoterList()` method. It retrieves the list of all voter details and returns it as JSON in the HTTP response.

7. `@GetMapping("/voterList/AllEligible")`: This annotation maps a GET request to "/auth/p2/voterList/AllEligible" to the `getAllEligibleVoters()` method. It retrieves the list of all eligible voter details and returns it as JSON in the HTTP response.

8. `@GetMapping("/voterList/{id}")`: This annotation maps a GET request with a path variable (denoted by `{id}`) to "/auth/p2/voterList/{id}" to the `getVoterListById()` method. It retrieves the voter details for the specified `id` and returns it as JSON in the HTTP response.

9. `@PostMapping("/voterList/insert")`: This annotation maps a POST request to "/auth/p2/voterList/insert" to the `voterListInsert()` method. It expects a JSON request body with

`VoterRegisterRequest` data, which is used to insert a new voter. The method returns a `ResponseEntity` containing `VoterAuthenticationResponse` data as JSON in the HTTP response.

10. `@PostMapping("/voterList/authenticate")`: This annotation maps a POST request to "/auth/p2/voterList/authenticate" to the `voterListAuthenticate()` method. It expects a JSON request body with `VoterAuthenticationRequest` data, which is used to authenticate a voter. The method returns a `ResponseEntity` containing `VoterAuthenticationResponse` data as JSON in the HTTP response.

In summary, the `VoterController` class defines endpoints for retrieving voter details, inserting new voters, and handling voter authentication. The actual implementation of the methods is expected to be in the `VoterService` and `VoterAuthenticationService` classes, which perform the business logic and data handling operations for voters and voter authentication, respectively. The controller handles HTTP requests and responses, and the service classes handle the underlying business logic.