Code Summary

### Summary  
  
#### Frontend (request.js)  
  
- \*\*API URL Configuration:\*\*  
 - `const API\_URL = "http://localhost:8000";` sets the base URL for the backend API.  
  
- \*\*Fetching Planets:\*\*  
 - `async function httpGetPlanets()` sends a GET request to the backend at `/planets` and returns the response in JSON format.  
  
#### Backend  
  
1. \*\*app.js (Express Application Setup)\*\*  
 - \*\*Module Imports:\*\*  
 - Importing necessary modules like `express`, `path`, `cors`, `morgan`, and the `planetsRouter`.  
  
 - \*\*Middleware Setup:\*\*  
 - Configures CORS to allow requests from `http://localhost:3000`.  
 - Uses `morgan` for logging requests.  
 - Uses built-in middleware to parse JSON bodies and serve static files from the `public` directory.  
  
 - \*\*Routing:\*\*  
 - Sets up the `planetsRouter` to handle requests to `/planets`.  
 - A catch-all route to serve the frontend application.  
  
2. \*\*server.js (Server Setup)\*\*  
 - \*\*Module Imports:\*\*  
 - Imports the HTTP module, the Express app, and the function to load planet data.  
  
 - \*\*Server Configuration:\*\*  
 - Sets the port to `8000` or the value in `process.env.PORT`.  
 - Creates an HTTP server with the Express app.  
  
 - \*\*Start Server:\*\*  
 - Asynchronously loads planet data before starting the server.  
 - Listens on the specified port and logs a message when the server starts.  
  
3. \*\*planets.router.js (Router Setup)\*\*  
 - \*\*Module Imports:\*\*  
 - Imports `express` and the `getAllPlanets` controller.  
  
 - \*\*Router Configuration:\*\*  
 - Creates an Express router.  
 - Defines a GET route for `/planets` that uses the `getAllPlanets` controller.  
 - Exports the router.  
  
4. \*\*planets.controller.js (Controller)\*\*  
 - \*\*Module Imports:\*\*  
 - Imports the planets data from the model.  
  
 - \*\*Controller Function:\*\*  
 - `getAllPlanets(req, res)` sends the list of planets as a JSON response with a status code of 200.  
 - Exports the controller function.  
  
5. \*\*planets.model.js (Model)\*\*  
 - \*\*Module Imports:\*\*  
 - Imports `path`, `csv-parse`, and `fs` modules.  
  
 - \*\*Data Handling:\*\*  
 - Defines an array `habitablePlanets` to store data.  
 - `isHabitablePlanet(planet)` checks if a planet meets criteria for habitability.  
  
 - \*\*Load Planets Data:\*\*  
 - `loadPlanetsData()` reads and parses the CSV file containing planet data.  
 - Filters and stores habitable planets in the `habitablePlanets` array.  
 - Resolves a promise when data loading is complete.  
  
 - \*\*Exports:\*\*  
 - Exports `loadPlanetsData` function and the `habitablePlanets` array as `planets`.  
  
### Interaction Flow  
  
1. \*\*Frontend Request:\*\*  
 - When the frontend calls `httpGetPlanets()`, it sends a request to `http://localhost:8000/planets`.  
  
2. \*\*Backend Response:\*\*  
 - The request is handled by the Express app in `app.js`, which routes it to the `planetsRouter`.  
  
3. \*\*Router Handling:\*\*  
 - The `planetsRouter` in `planets.router.js` directs the request to the `getAllPlanets` controller.  
  
4. \*\*Controller Response:\*\*  
 - The `getAllPlanets` controller in `planets.controller.js` fetches the data from the `planets.model.js` and sends it back as a JSON response.  
  
5. \*\*Data Loading:\*\*  
 - The `loadPlanetsData` function in `planets.model.js` reads planet data from a CSV file, filters for habitable planets, and stores them in an array used by the controller.